

ARHITECTURA SISTEMELOR DE CALCUL - CURS 0x06

ARHITECTURA CALCULATOARELOR MODERNE

Cristian Rusu

CUPRINS

- **arhitectura de bază a calculatoarelor**
- **câteva detalii despre fiecare componentă hardware ...**

STRUCTURA CURSULUI - UNDE SUNTEM

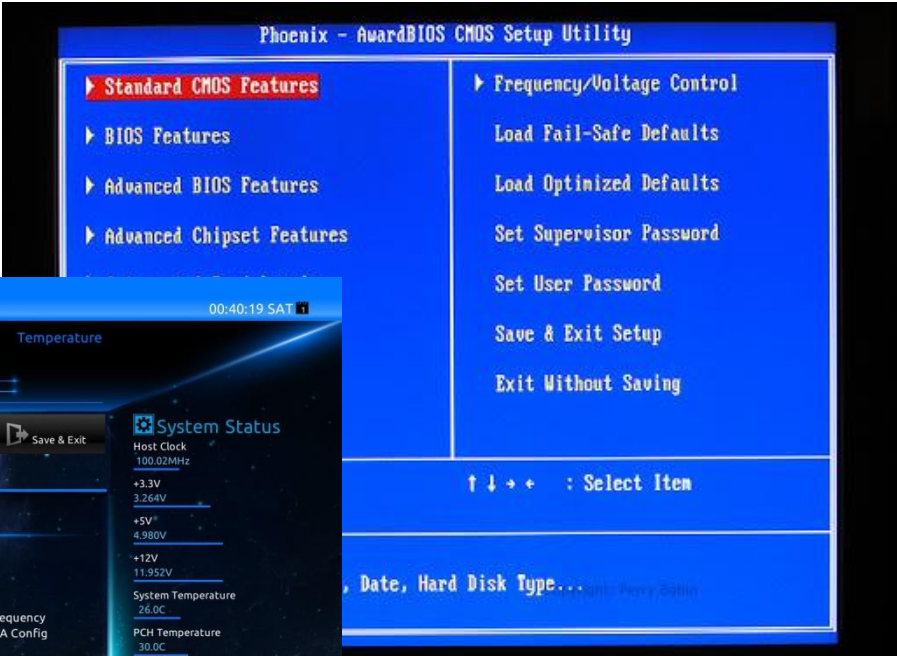
- **circuite digitale**
 - teoria informației și abstractizarea digitală
 - funcții și circuite logice
- **arhitecturi de calcul**
 - seturi de instrucțiuni
 - limbajul assembly
 - compilatoare
 - pipelining
 - ierarhia memoriei
- **organizarea calculatoarelor**
 - unitatea de procesare centrală
 - performanța calculatoarelor
 - dispozitive periferice și întreruperi
 - calcul paralel

Operații de bază cu sisteme de calcul

- pornirea sistemului
 - în general, un buton de “power on” / “power off” cum funcționează un astfel de buton? atât la pornire cât și la oprire?
 - realizează alimentarea cu electricitate a componentelor
 - **CPU** este activat
 - CPU caută/pornește **BIOS** (Basic Input Output System)
 - testează componentele hardware (RAM, I/O, HD, etc.)
 - BIOS este scris în **ROM** (Read Only Memory) pe placa de bază
 - este scris într-un tip de memorie nevolatilă
 - pentru execuție, BIOS-ul este încărcat în RAM
 - BIOS știe cât e ceasul (**CMOS Real-Time Clock**) și hardware-ul
 - îl accesați automat când porniți calculatorul, fie cu F2 (în general)
 - CPU/BIOS pornesc Boot Code (caută sistemul de operare)
 - sistemul de operare este în general pe HD (poate fi și pe CD, stick)
 - sistemul de operare este încărcat în RAM pentru execuție

Operații de bază cu sisteme de calcul

- BIOS/UEFI



PhoenixBIOS Setup Utility

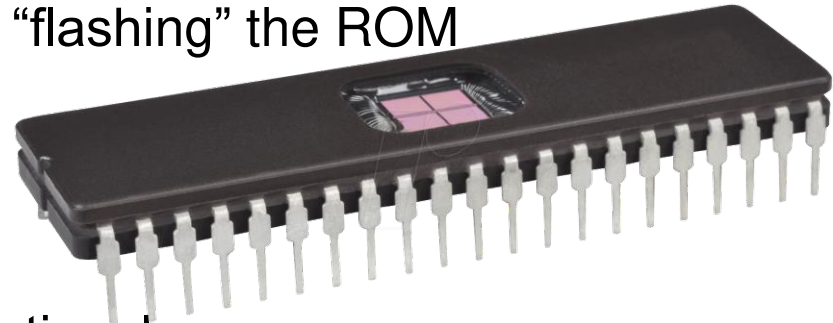
Utility	Boot	Exit
[09:21:30] [09/02/2016]	[1.44/1.25 MB 3 1/2"] [Disabled]	Item Specific Help <Tab>, <Shift-Tab>, or <Enter> selects field.
[None]	[None]	
[CD-ROM]	[None]	
Secondary Slave		
Keyboard Features		
System Memory:	640 KB	
Extended Memory:	2096128 KB	
Boot-time Diagnostic Screen:	[Enabled]	

F1 Help ↑ Select Item -/+ Change Values F9 Setup Defaults
Esc Exit ↔ Select Menu Enter Select Sub-Menu F10 Save and Exit

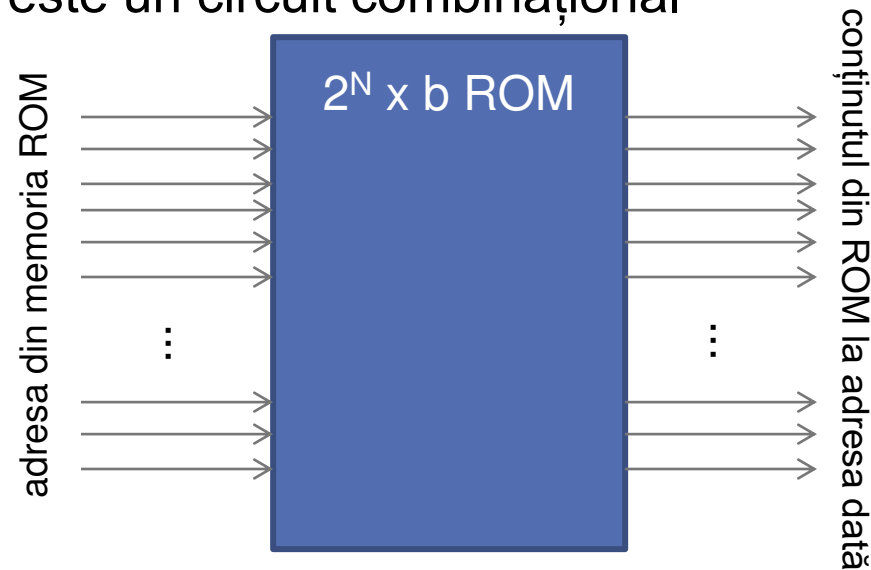
Operații de bază cu sisteme de calcul

- BIOS

- este scris în ROM (Read Only Memory)
- câteodată în Programmable ROM / Erasable Programmable ROM / Electrically Erasable Programmable ROM
- să scriem în ROM: “burning” sau “flashing” the ROM
- este un tip de *firmware*



- în general este un circuit combinațional



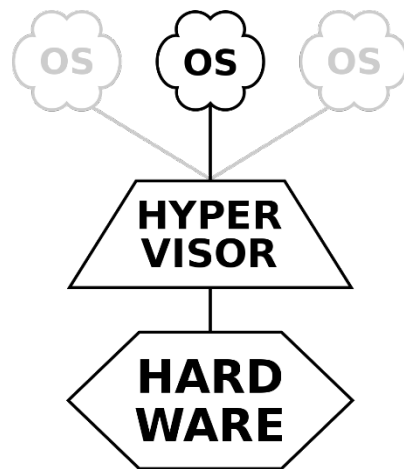
Operații de bază cu sisteme de calcul

- mai multe sisteme de operare pe același sistem de calcul

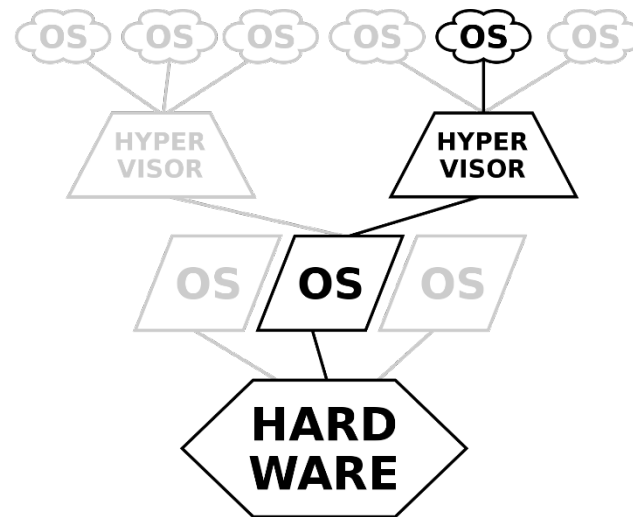


Operații de bază cu sisteme de calcul

- OS-ul preia controlul de la BIOS
 - din acest moment doar OS-ul are acces direct la periferice
 - accesul este realizat prin *drivere*
- virtualizare (hardware)
- emulare



TYPE 1
native
(bare metal)



TYPE 2
hosted

- containere (dockere)

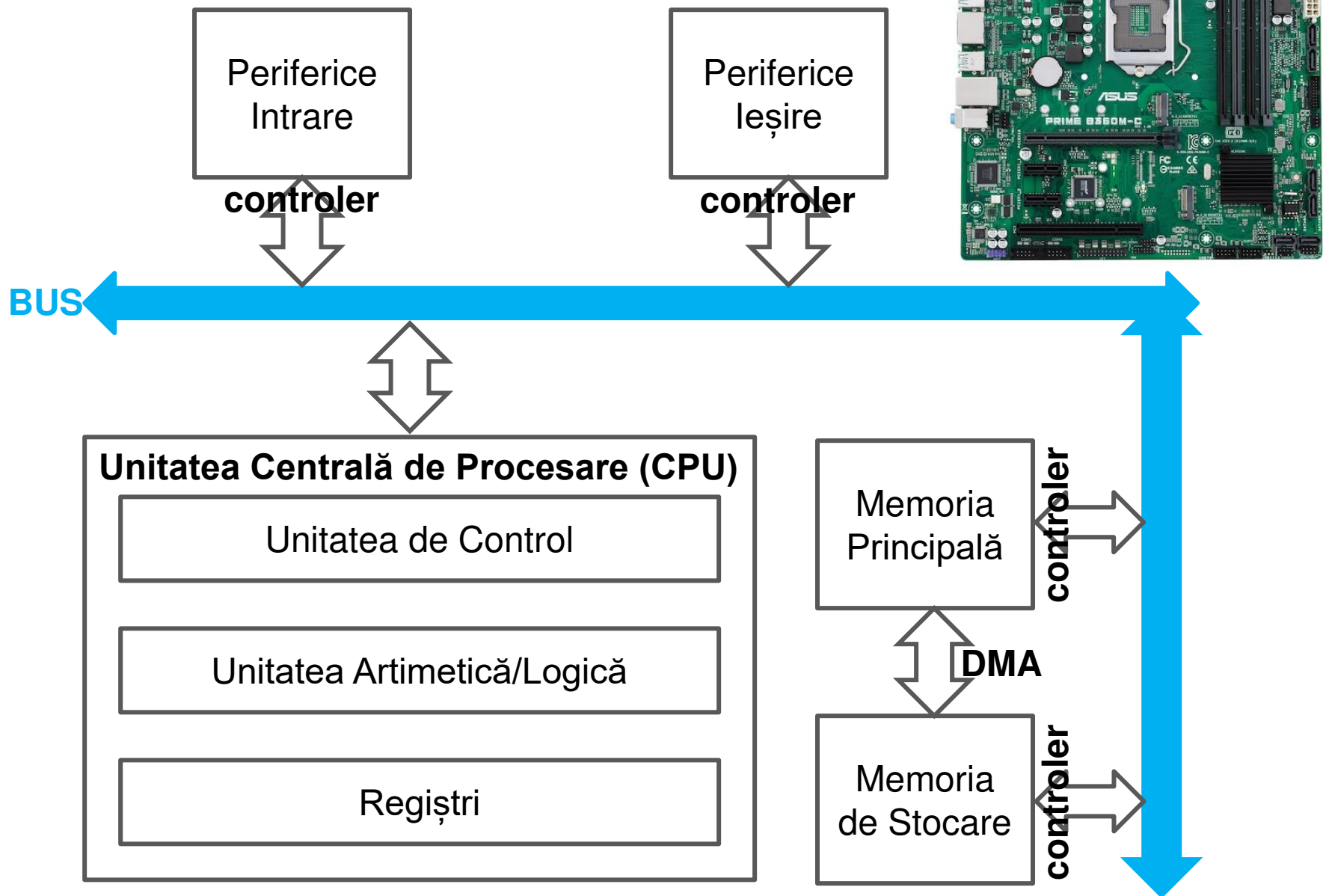
Operații de bază cu sisteme de calcul

- **OS-ul preia controlul de la BIOS**
 - din acest moment doar OS-ul are acces direct la periferice
 - accesul este realizat prin *drivere*
 - OS-ul oferă o imagine abstractizată a memoriei pentru fiecare proces pornit
- din momentul în care OS-ul pornește, sistemul de calcul intră în ciclul obișnuit de procesare (secvența de boot s-a terminat)

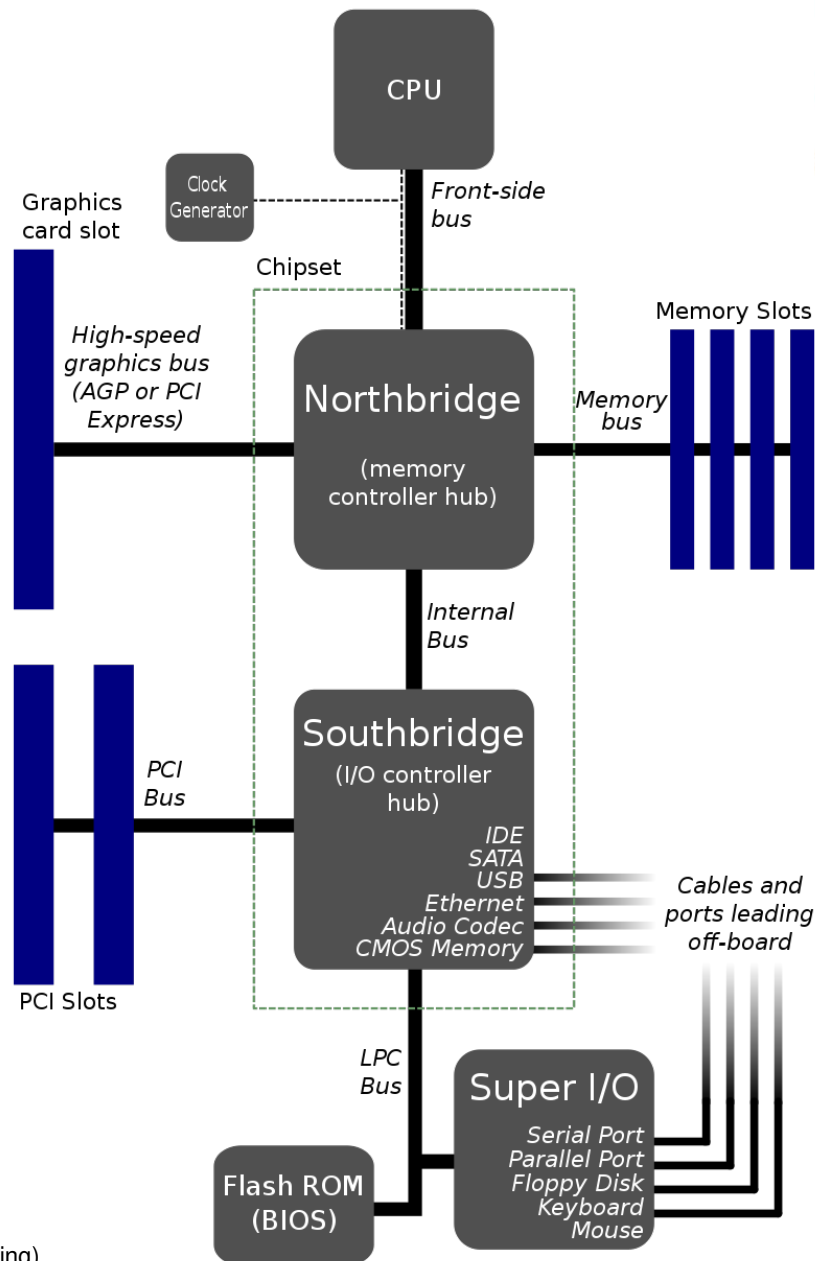
ARHITECTURA DE BAZĂ

- **un sistem de calcul trebuie să fie capabil:**
 - să calculeze
 - să execute instrucțiuni
 - să comunice
 - să transfere biți între componente electronice
 - să stocheze
 - date care să fie folosite de instrucțiuni
 - instrucțiuni pentru execuție

ARHITECTURA DE BAZĂ



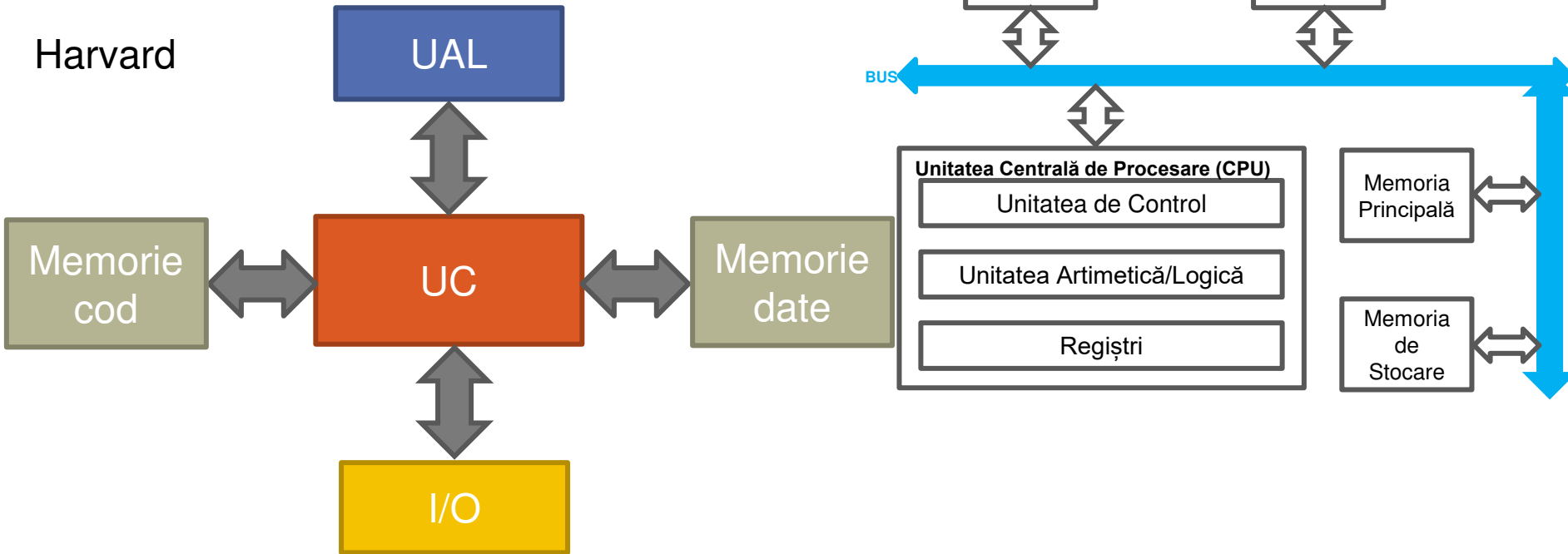
ARHITECTURA DE BAZĂ



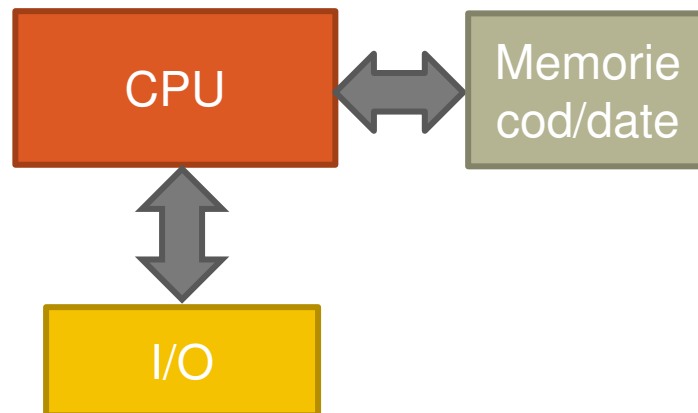
ARHITECTURA DE BAZĂ

- Tipul arhitecturii de calcul

Harvard



von Neumann



calculatoarele recente încep
să nu mai fie von Neumann

ARHITECTURA DE BAZĂ

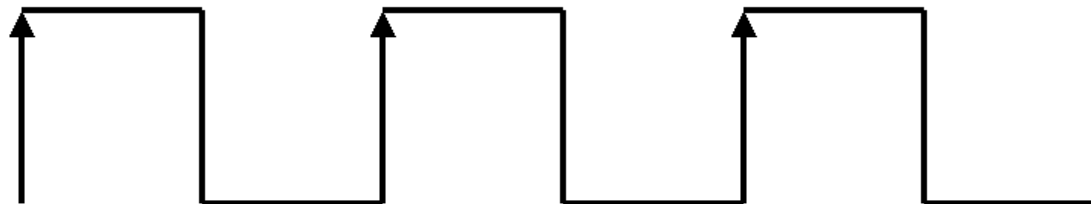
- **Unitatea Centrală de Procesare**

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni

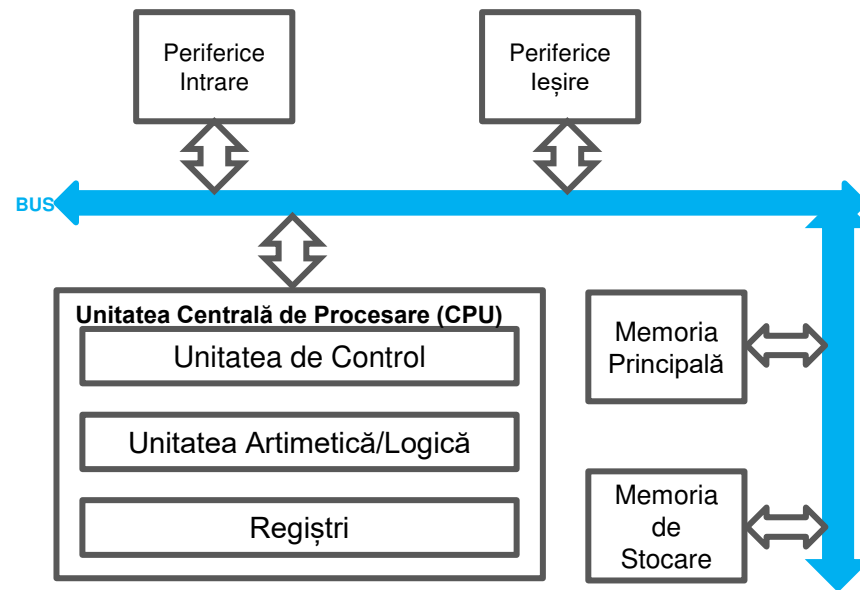
- 5 componente principale:

- **Clock**

- este un circuit special care generează “ceasul”



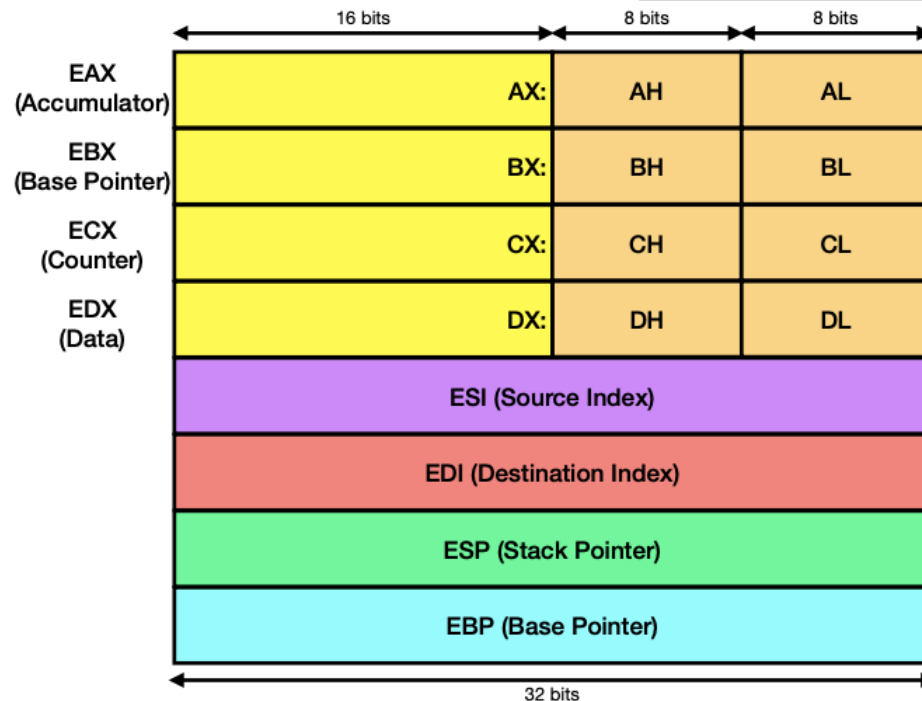
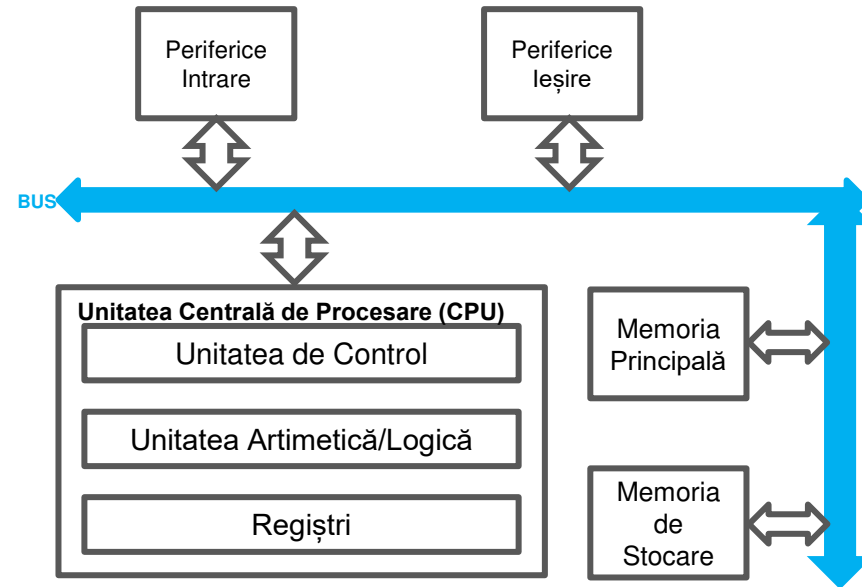
- este frecvența la care operează (calculare și sincronizarea componentelor secvențiale) CPU-ul
 - cu cât este mai mare frecvența, cu atât mai bine (în general)
 - se măsoară în MHz sau GHz



ARHITECTURA DE BAZĂ

- Unitatea Centrală de Procesare

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni
- 5 componente principale:
 - **regiștri (“memoria”)**



ARHITECTURA DE BAZĂ

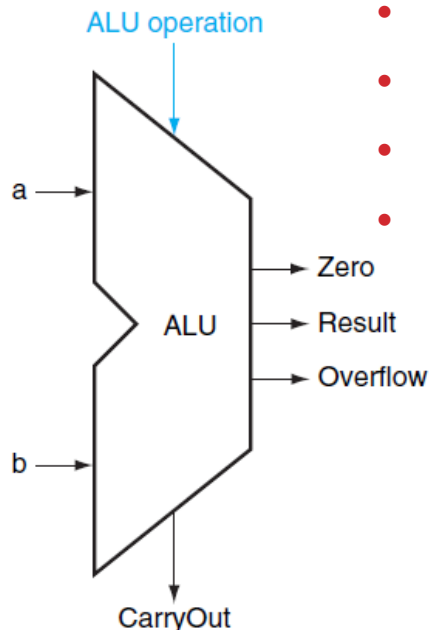
- **Unitatea Centrală de Procesare**

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni

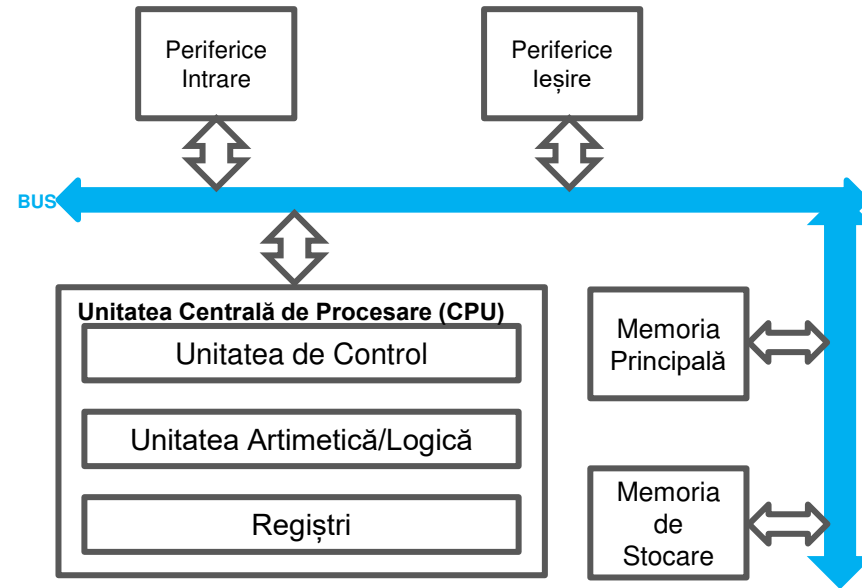
- 5 componente principale:

- **UAL (“operații”)**

- operații aritmetice cu întregi
 - operații logice
 - operații aritmetice cu numere în formatul floating point
 - operații speciale: sqrt, exp, trig



după orice operație, știm
“gratuit” dacă rezultatul a fost
sau nu zero – este folositor?



ARHITECTURA DE BAZĂ

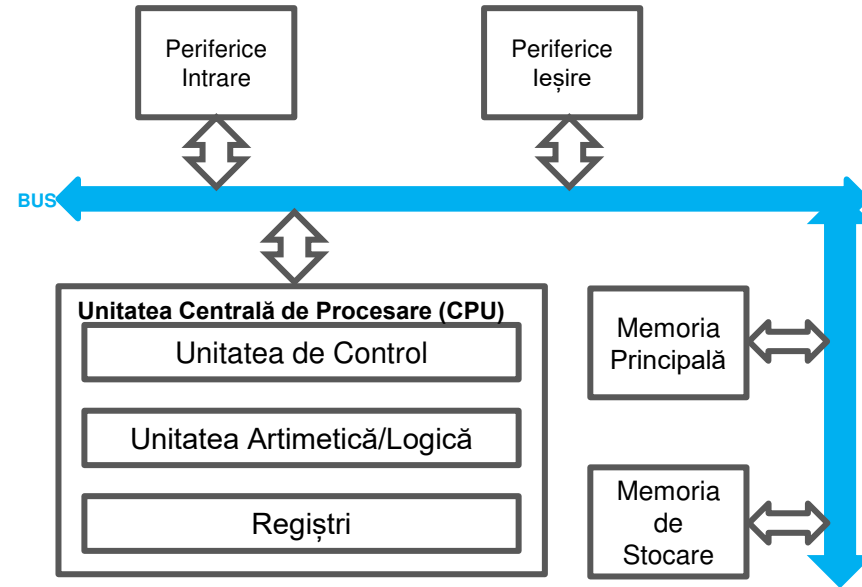
- **Unitatea Centrală de Procesare**

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni

- 5 componente principale:

- **BUS**

- CPU are nevoie de șiruri de biți din memoria principală sau cea de stocare
 - CPU are nevoie să scrie înapoi în memorie rezultate
 - CPU coordonează perifericele



ARHITECTURA DE BAZĂ

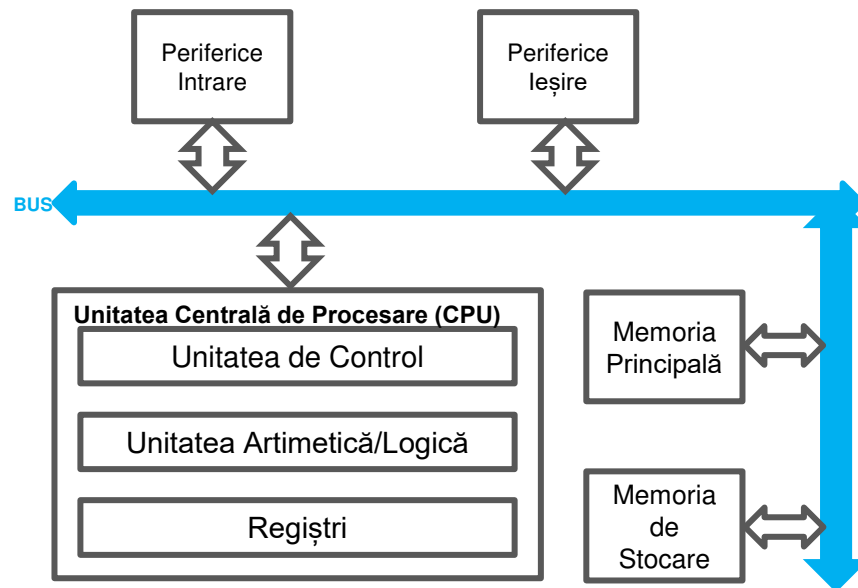
- **Unitatea Centrală de Procesare**

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni

- 5 componente principale:

- **UC (“instrucțiunile”)**

- fetch
 - citim din memorie codul care trebuie executat
 - de unde din memorie? Instruction Pointer ne spune
 - decode
 - circuitul “Instruction Decoder” analizează biții citați din memorie ca să “înțeleagă” ce să facă cu ei
 - execute
 - execută instrucțiunea decodată
 - poate duce la schimbarea IP sau la transmiterea ceva pe BUS către memorie
 - calculează următorul IP



ARHITECTURA DE BAZĂ

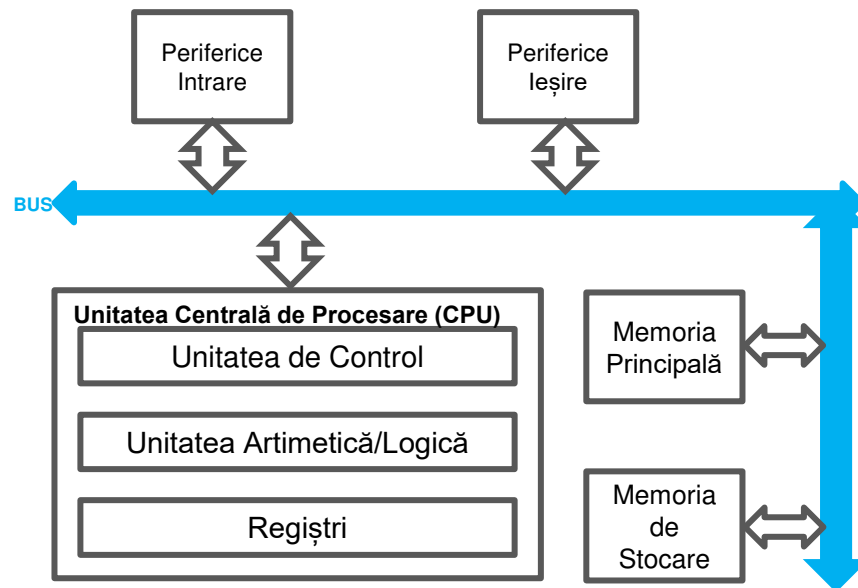
- **Unitatea Centrală de Procesare**

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni

- 5 componente principale:

- **UC (“instrucțiunile”)**

- fetch
 - IP = 10011 (locația în memorie de unde să citim biții)
 - după citire, IP este actualizat
 - decode
 - s-a citit “11000110” care este decodat în
 - opcode = 110, operand1 = 00 operand2 = 110
 - de exemplu: 110 = “adună valoarea imediată A la registrul R”, R = 00 este EAX (prin convenție), A = 110 (adică 6)
 - execute
 - trimite $EAX \leftarrow EAX + 6$ la UAL
 - citește rezultatul din UAL și pune-l în registrul EAX



ARHITECTURA DE BAZĂ

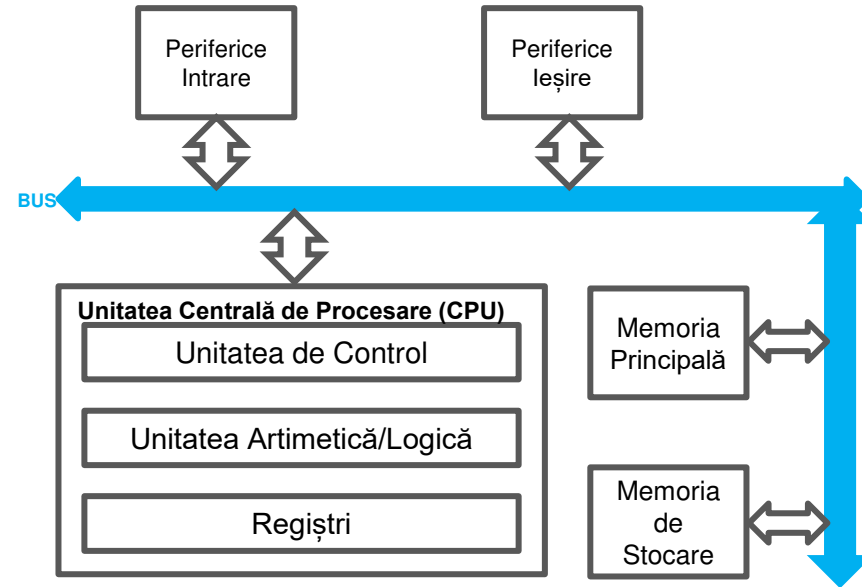
- **Unitatea Centrală de Procesare**

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni

- 5 componente principale:

- **UC (“instrucțiunile”)**

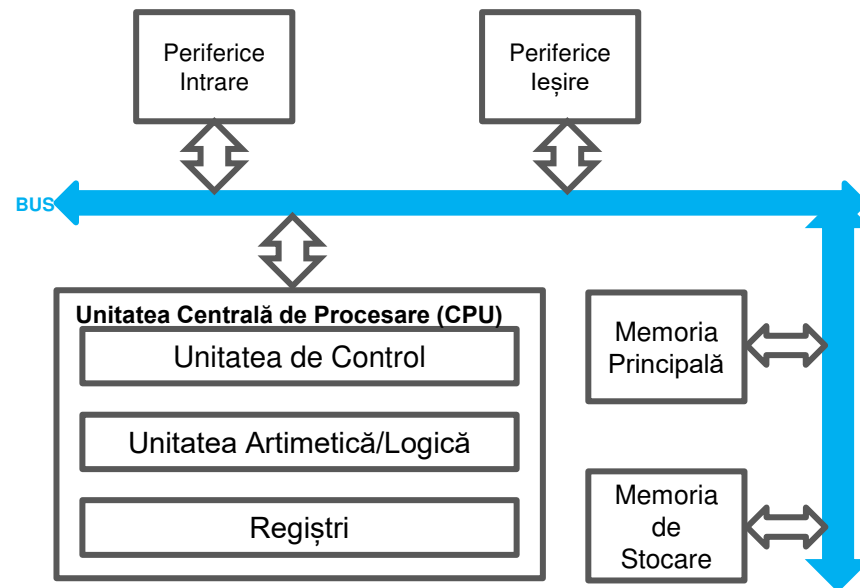
- fetch
 - IP = 10011 (locația în memorie de unde să citim biții)
 - după citire, IP este actualizat
 - decode
 - s-a citit “1110011” care este decodat în
 - opcode = 111, operand1 = 00 operand2 = 11
 - de exemplu: 111 = “adună registrul A la registrul R”, R = 00 este EAX, A = 11 este EDX (prin convenție)
 - execute
 - trimite $EAX \leftarrow EAX + EDX$ la UAL
 - citește rezultatul din UAL și pune-l în registrul EAX



ARHITECTURA DE BAZĂ

- **Unitatea Centrală de Procesare**

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni
- 5 componente principale:
 - **UC (“instrucțiunile”)**



IF – Instruction Fetch (citirea din memorie a instrucțiunilor)
ID – Instruction Decode (circuit secvențial care decodează)
EX – Execute (execuția propriu-zisă)
MEM – Memory Access (orice access memorie)
WB – Write Back (scrie rezultatul înapoi în memorie)

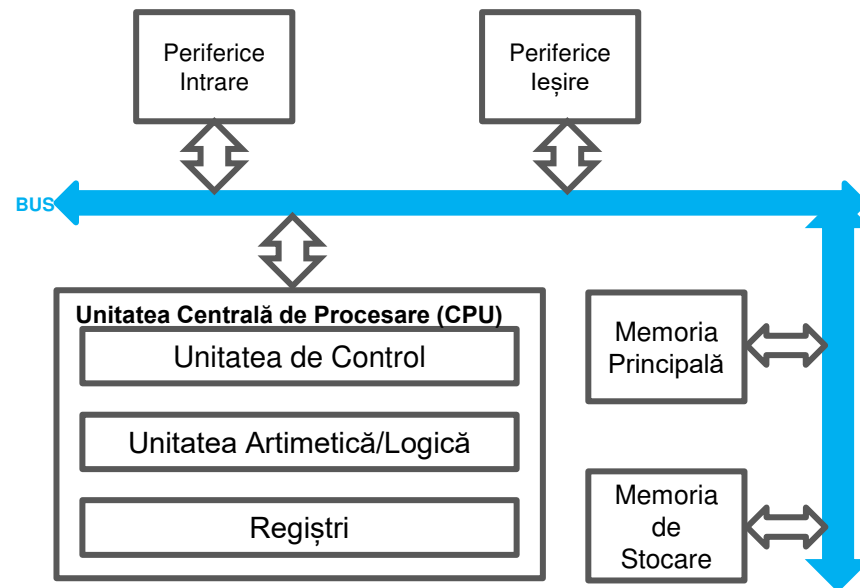
ARHITECTURA DE BAZĂ

- **Unitatea Centrală de Procesare**

- a.k.a. CPU
- este “creierul” unității de calcul
- execută instrucțiuni

- 5 componente principale:

- **Clock**
- **registri (“memoria”)**
- **UAL (“operații”)**
- **BUS**
- **UC (“instrucțiunile”)**



PROCESOR

Producator procesor	Intel®
Tip procesor	i9
Model procesor	9880H
Arhitectura	Coffee Lake
Numar nuclee	8
Frecventa nominala	2.3 GHz
Cache	16384 KB
Frecventa Turbo Boost	4.8 GHz
Tehnologie procesor	14 nm
Procesor grafic integrat	Intel® UHD Graphics 630

ARHITECTURA DE BAZĂ

- **Memoria Principală**

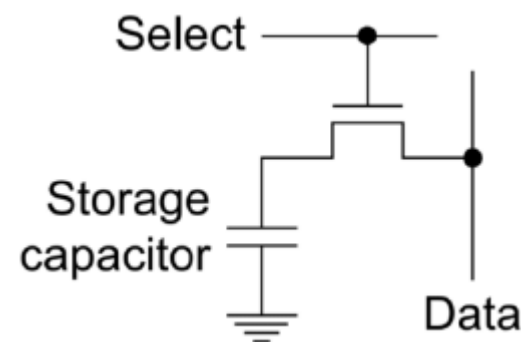
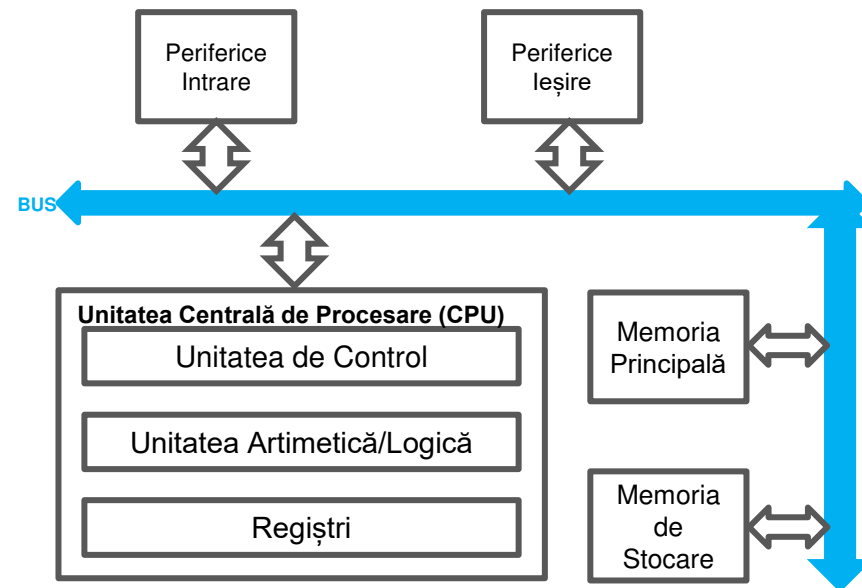
- conține cod și date
- este volatilă

- **Static RAM (SRAM)**

- bazată pe flip-flops
- rapid
- scump
- regiștrii din CPU sunt de același tip

- **Dynamic RAM (DRAM)**

- fiecare bit este reprezentat de o combinație tranzistor + condensator
- condensatoarele suferă de leakage (scurgeri de tensiune)
- DRAM trebuie actualizat o dată la fiecare câteva zeci de ms



de ce este DRAM mai ieftin decât SRAM?

are DRAM niște dezavantaje în comparație cu SRAM?

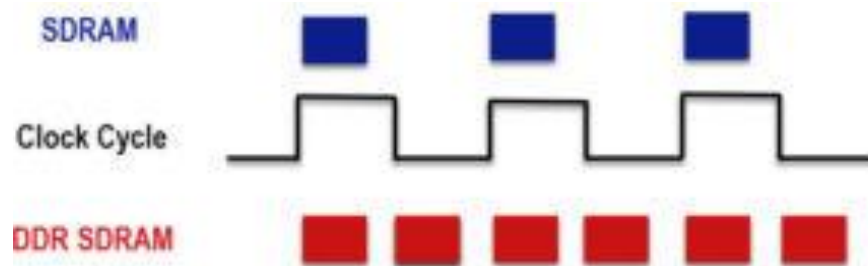
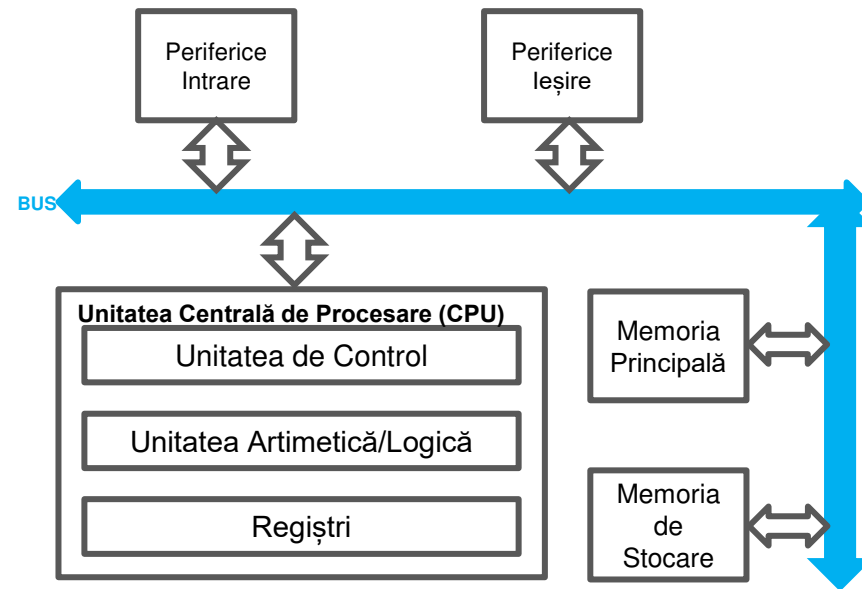
ARHITECTURA DE BAZĂ

- **Memoria Principală**

- conține cod și date
- este volatilă

- **DDR RAM**

- **Double Data Rate RAM**
- .../DDR4/DDR5/DDR6
- performanța este definită de:
 - capacitate
 - dacă au un sistem intern de corectarea erorilor (ECC)
 - timpi de acces (în cât timp de la comanda de citire de biți din RAM avem datele disponibile?, timpul de refresh)
 - consumul de energie

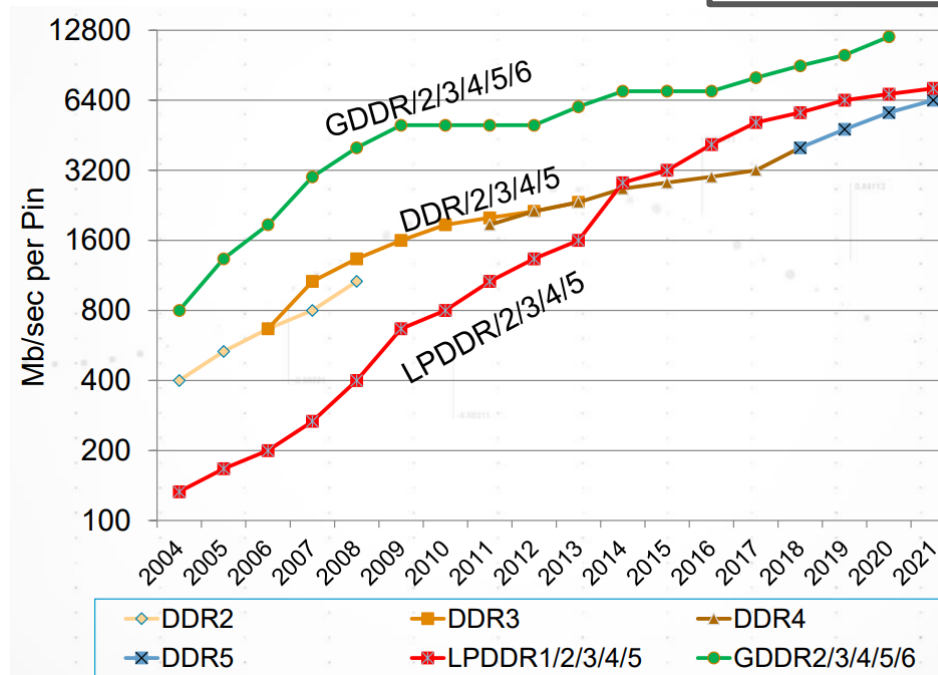
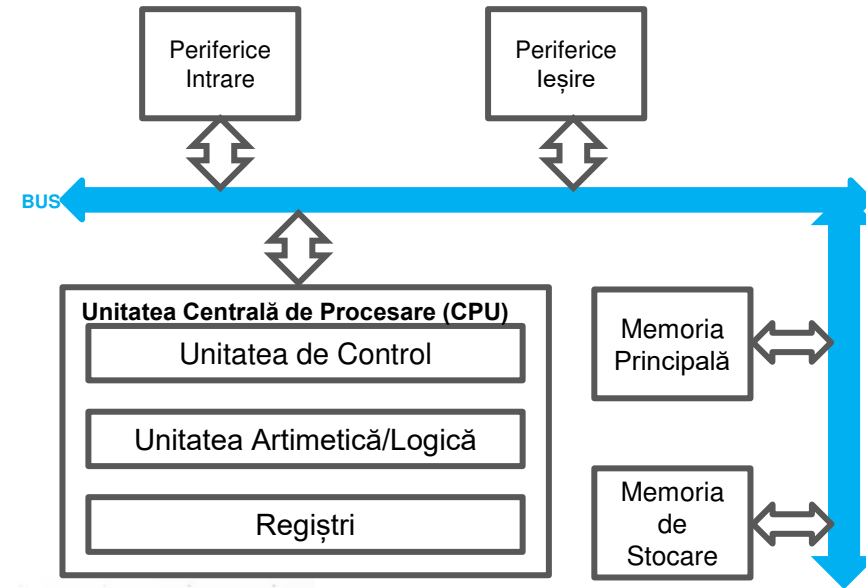


ARHITECTURA DE BAZĂ

- Memoria Principală

- conține cod și date
- este volatilă

- ce avem azi?

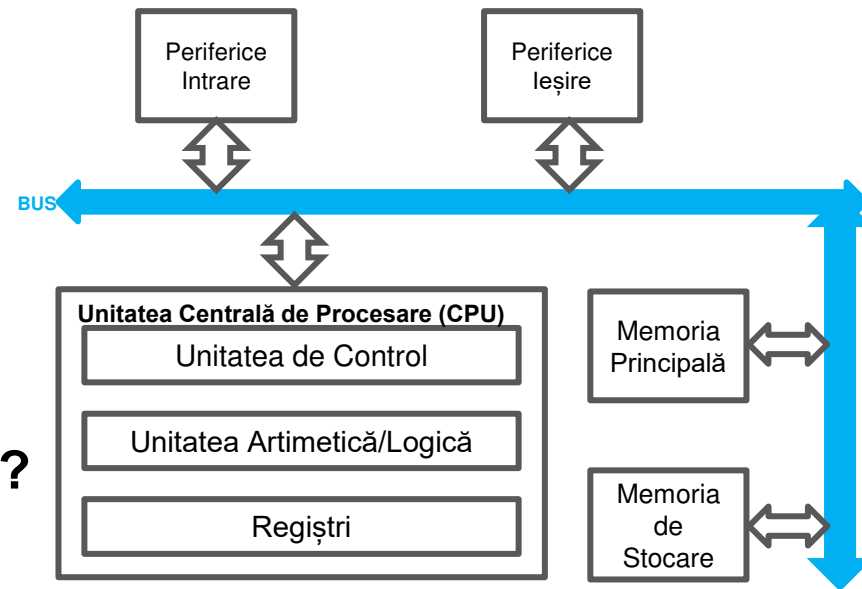


ARHITECTURA DE BAZĂ

- **Memoria Principală**

- **conține cod și date**
- **este volatilă**

- **ce ne interesează la memorie?**



MEMORIE

Capacitate memorie	32 GB
Tip memorie	DDR4
Numar sloturi	4
Sloturi ocupate	2
Frecventa	2666 MHz
Capacitate memorie maxima suportata	128 GB

ARHITECTURA DE BAZĂ

- **Memoria de Stocare**

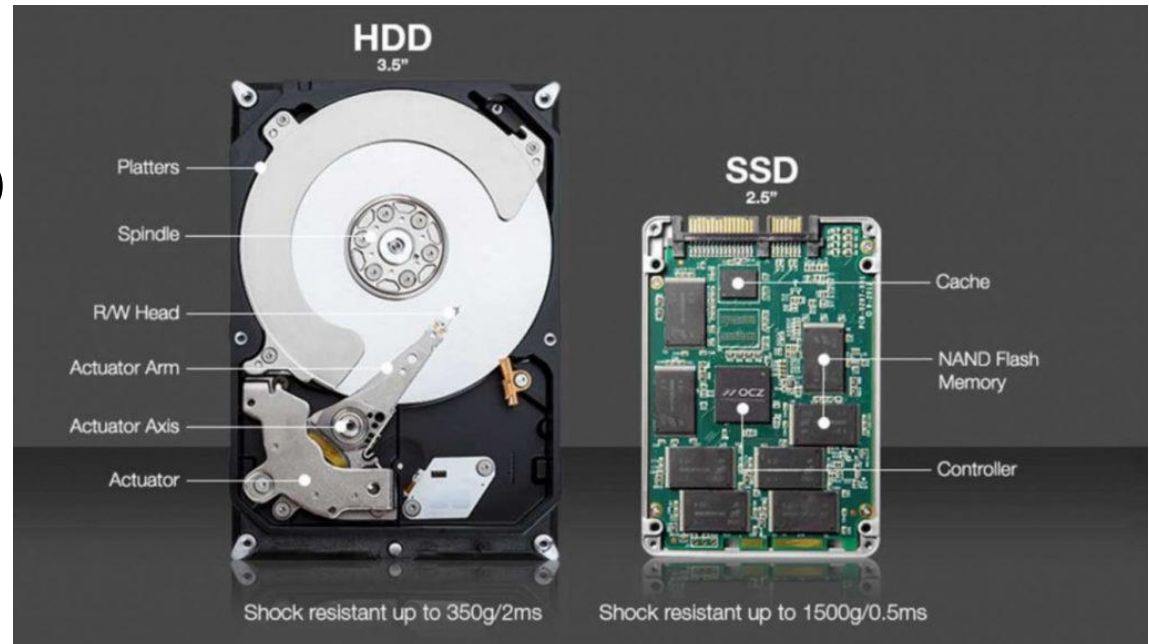
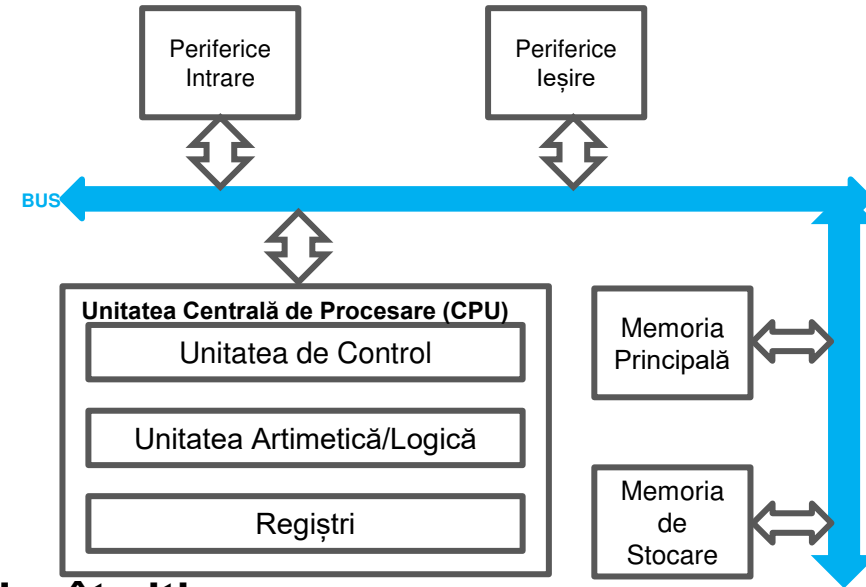
- conține cod și date
- este nevolatilă

- **SSD (Solid State Disks)**

- e memorie flash, rapidă
- azi, e scumpă
- scrierea e mult mai lentă decât citirea

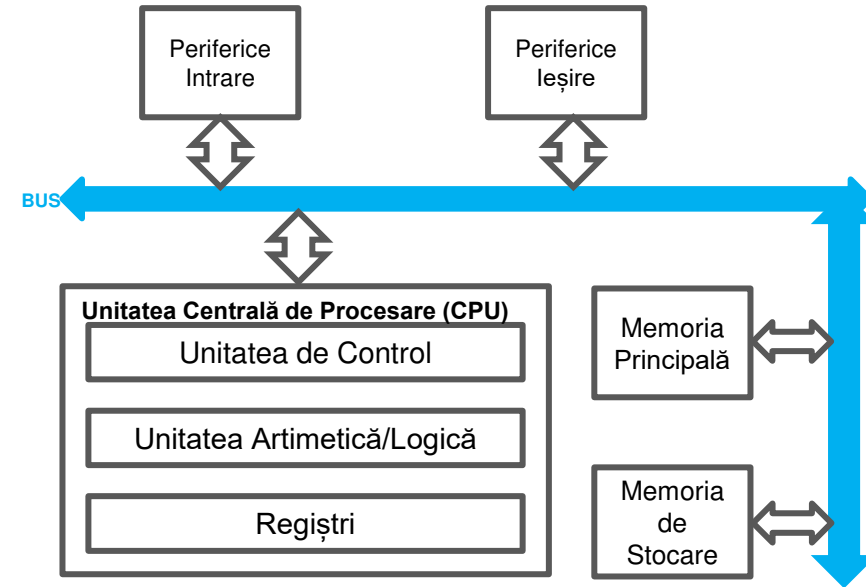
- **HDD (Hard Disks)**

- mecanic

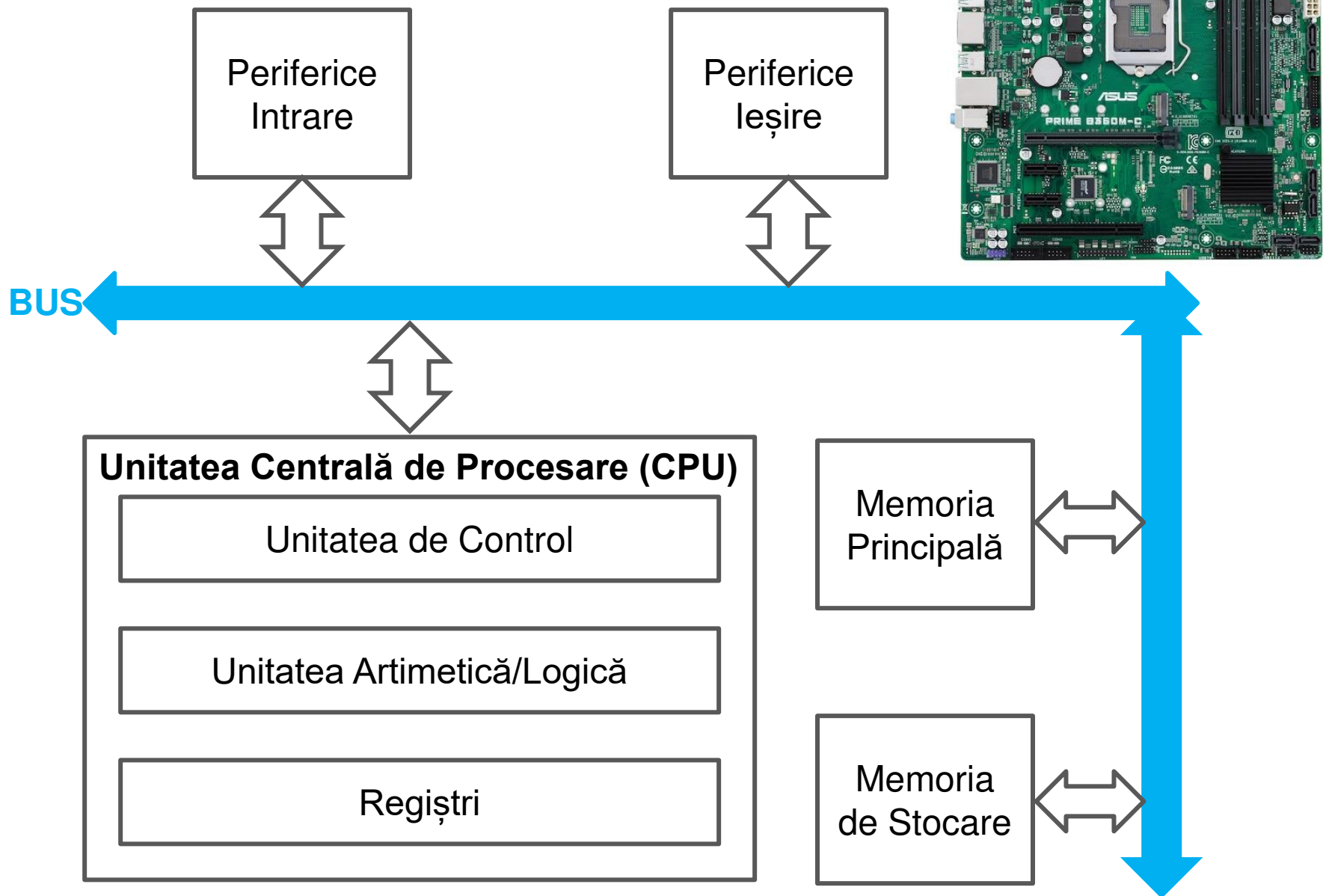


ARHITECTURA DE BAZĂ

- **BUS**
 - conectează CPU/memorie
 - proprietăți
 - capacitatea (bandwidth)
 - viteza (MHz)



ARHITECTURA DE BAZĂ



un astfel de sistem poate executa *doar cod mașină* pentru un *singur stream*